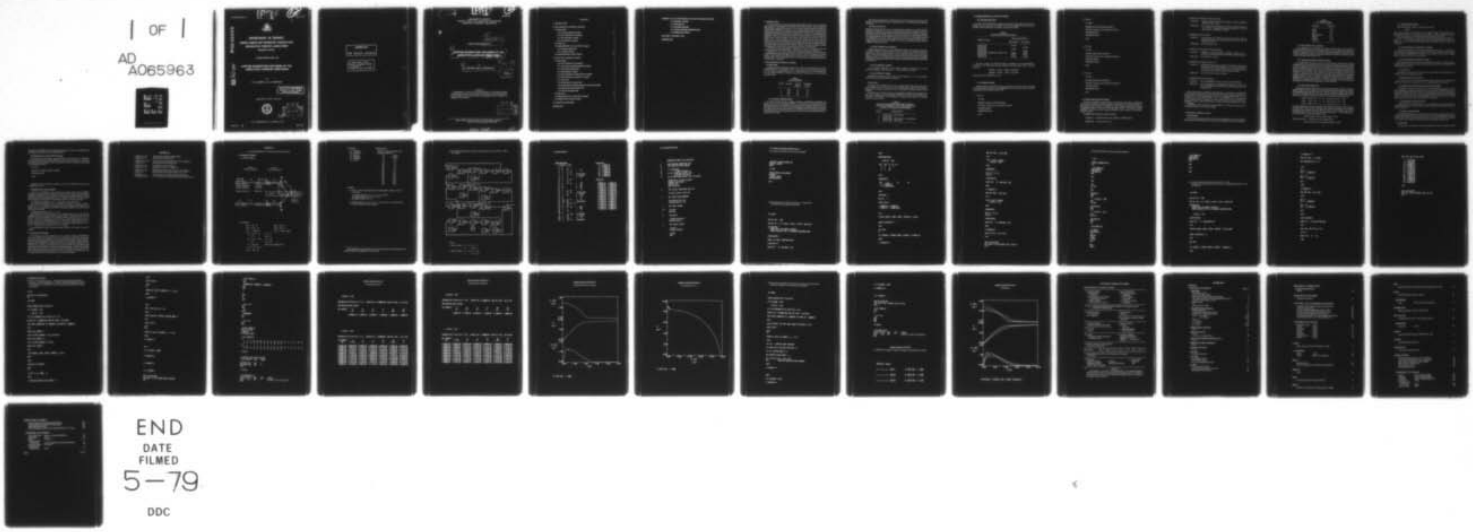
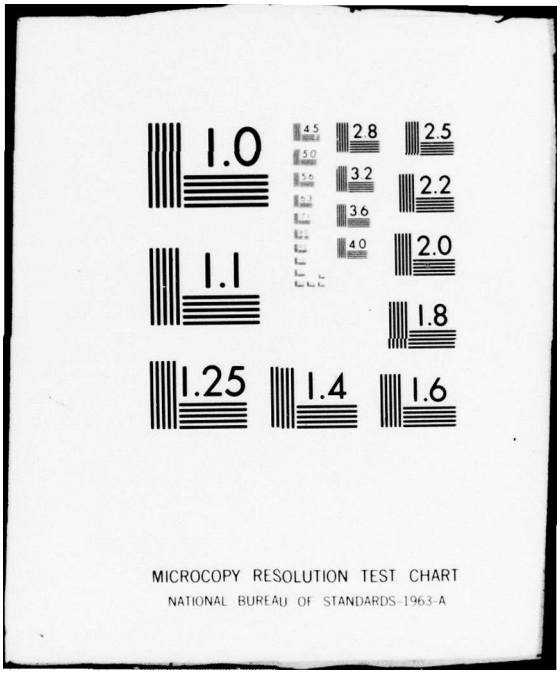
AD-A065 963 AERONAUTICAL RESEARCH LABS MELBOURNE (AUSTRALIA) F/G 9/2
FURTHER INFORMATION FOR USERS OF THE SIMULATION LANGUAGE CSMP-1--ETC(U)
MAY 78 N E GILBERT, P G NANKIVELL

UNCLASSIFIED ARL/AERO NOTE-375 NL

1 OF 1
AD A065963



END
DATE FILMED
5-79
DDC


1.0
1.1
1.25
1.4
1.6
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.8
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

ARL-AERO-NOTE-375

LEVEL II

12

AR-001-267

DEPARTMENT OF DEFENCE

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION

AERONAUTICAL RESEARCH LABORATORIES

MELBOURNE, VICTORIA

AERODYNAMICS NOTE 375

FURTHER INFORMATION FOR USERS OF THE SIMULATION LANGUAGE CSMP-10(ARL)

by

N. E. GILBERT and P. G. NANKIVELL

Approved for Public Release

COPY No 9

MAY 1978

AR-001-267

DEPARTMENT OF DEFENCE
DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION
AERONAUTICAL RESEARCH LABORATORIES

AERODYNAMICS NOTE-375

FURTHER INFORMATION FOR USERS OF THE SIMULATION LANGUAGE CSMP-10(ARL)

by

N. E. GILBERT and P. G. NANKIVELL

11 May '78

12 40p.

SUMMARY

This document is a postscript to previous publications on the simulation language CSMP-10(ARL). It consists of a description of improvements, details of the location of source files and their use on the ARL computer, and suggestions on how to implement the language on other computers.

POSTAL ADDRESS: Chief Superintendent, Aeronautical Research Laboratories,
Box 4331, P.O., Melbourne, Victoria, 3001, Australia.

CONTENTS

1. INTRODUCTION

Since documenting the simulation language CSMP-10(ARL) (see Ref. 1 and 2), there have been a number of developments necessitating the following postscript. These developments, described below, include (a) the provision of an alternative integration method, (b) compilation using the DEC FORTRAN-10 compiler, and (c) a number of improvements, especially to the output program. Some of these developments are included in Reference 3, but are only described very briefly. Developments (a) and (b) have resulted in four versions of the modelling program BOMMP. In order to avoid unnecessary repetition of stored information, the source files have therefore been suitably reorganized. The location of the necessary source files at ARL and details on how to obtain running versions on the ARL computer are given here. Following interest shown in using the language on other computers, the uses made of a small number of subroutines written in the DEC machine language MACRO-10 have been investigated. Descriptions of the operation performed by each of these routines are therefore provided here together with suggestions on action to take when alternative subroutines are unavailable.

The aircraft arresting gear problem of Reference 4 is used to fully demonstrate the new operating procedures and improvements to the simulation language (see Appendix A).

In References 1 and 2, when showing terminal messages, contrasting type faces were used to distinguish between messages typed by the user and those typed by the appropriate computer program. Once familiar with CSMP-10(ARL), the above style of presentation becomes unnecessary. Hence, to avoid high typesetting costs for this publication, the actual terminal printout is used directly in Appendix A.

2. THE MODELLING PROGRAM 'BOMMP'

2.1 Improvements

The improvements described below in Sections 2.1.1 and 2.1.2 have resulted in four versions of BOMMP which are summarized in Table 1. It should be noted that Versions 2 and 4 may also be compiled using the FORTRAN IV (i.e. F40) compiler as well as the FORTRAN-10 (i.e. F10) compiler. Whichever compiler is used however, Versions 2 and 4 do not provide expandable arrays.

TABLE 1
Versions of BOMMP

Version	Compiler	Adams Integrator	Expandable Arrays
1	F40	no	yes
2	F10	no	no
3	F40	yes	yes
4	F10	yes	no

2.1.1 Alternative Integration Method

An Adams predictor-corrector integration method, as implemented by Shampine and Gordon (see Ref. 5), has been made available as an alternative to the standard second-order Runge-Kutta method used in BOMMP. The Adams method automatically optimizes the order and integration interval to provide maximum computing speed for specified accuracy and round-off error. This method provides large reductions in computing time when solving non-stiff systems of differential equations in which significant periods of inactivity occur and in which the derivatives are expensive to compute.

The second-order Runge-Kutta method may be used in all versions. For the versions that include the Adams integrator, there are slight differences to the 'INTEGRATION' command. The message

ADAMS INTEGRATOR:

appears first, and the user must type "Y" to specify the Adams integrator, or any other character (including a 'carriage-return') to specify the Runge-Kutta second order integrator. The rest of the prompts should be answered as before. When the Adams integrator is specified, the integration interval is ignored. The Adams integrator automatically varies its order and integration interval for maximum efficiency within specified relative and absolute errors. These may be specified by including them, in the above order suitable for input via the statement FORMAT (2F), in a file on logical unit 37 called ADAMS.ERR. If this file is not provided, the values 0·0001 and 0·0000001 respectively are used.

2.1.2 DEC FORTRAN-10 Compilation

Minor changes have been made to the FORTRAN coding to allow compilation using the F10 compiler, which can produce programs that run up to 40% more efficiently in CPU time. However, no routine is presently available to provide expandable arrays in F10 compiled programs. In F10 versions therefore, there is a maximum block number of 300 and a maximum number of I and T1, U, and F blocks of 100, 25, and 25 respectively. These limits may be altered by changing the appropriate DIMENSION statements in the files MAIN2.FOR or MAIN4.FOR (see Section 2.2).

2.1.3 The 'RESUME' Command

A new command, 'RESUME', has been added. It enables the continuation of a run that has been interrupted by either typing an "↑", or the execution of a Q block.

2.1.4 The 'RETAIN' Command

The 'RETAIN' command previously reset the initial condition of I, T1, U, and Z blocks. It now also resets the initial condition of UO blocks.

2.2 Source Files

The source files for BOMMP may be found on DECtape 068 by ARL users. Copies will be made available to others on request when a 7-track magnetic tape is supplied. Unless otherwise requested, the code will be recorded at 556 b.p.i. in 6-bit BCD code with one 84-byte record per block.

Files with extensions 'FOR' and 'MAC' are FORTRAN and MACRO-10 coded respectively. Each version requires the files CSMPA.FOR, CSMPB.FOR, CSMPC.FOR, FCHECK.MAC, CPU.MAC, TTYCHK.MAC, and DUSER.FOR. Additional files required for individual versions are given in Table 2.

TABLE 2
For each version of BOMMP, files required in addition to CSMPA.FOR, CSMPB.FOR, CSMPC.FOR, FCHECK.MAC, CPU.MAC, TTYCHK.MAC, and DUSER.FOR

Version	Additional Files
1	MAIN1.FOR, EXPAND.MAC
2	MAIN2.FOR
3	MAIN3.FOR, EXPAND.MAC, INTEG.FOR
4	MAIN4.FOR, INTEG.FOR

2.3 Running BOMMP on the ARL PDP-10 Computer

2.3.1 Relocatable Binary Files

For users on the ARL PDP-10 computer, the relocatable binary files may be found on DECtape 137. The source files, the compiler used, and the subsequent 'REL' files are given in Table 3. Where an extension is not shown, it is assumed to be 'REL'.

TABLE 3
Relocatable binary files and their sources

FORTRAN Source	Relocatable binary files	
	F40 Compiler	F10 Compiler
MAIN1.FOR	MAIN1	—
MAIN2.FOR	—	MAIN2
MAIN3.FOR	MAIN3	—
MAIN4.FOR	—	MAIN4
CSMPA.FOR, CSMPB.FOR, CSMPC.FOR	SUB40	SUB10
INTEG.FOR	INT40	INT10
DUSER.FOR	DUS40	DUS10

The 'REL' versions of the MACRO routines are combined in the file MACROS.REL. The Version 1 and 2 'REL' files have been combined as follows (extension 'REL' is assumed as above)

$$\text{BOM40} = \text{MAIN1} + \text{SUB40} + \text{MACROS}$$

$$\text{BOMMP} = \text{MAIN2} + \text{SUB10} + \text{MACROS}$$

and may be found on DSKB: [1031, 1161].

2.3.2 Loading the Progrhm

Assuming that the required 'REL' files are on the user's own disk area, then a core image of the program may be produced in the file BOMMP.SAV for each version as follows. The files are loaded by using the system program 'LINK'.

a) Version 1

```
. R LINK
*BOM40 (or MAIN1, SUB40, MACROS)
*(the 'REL' versions of any user-defined subroutines)
*/SEARCH DUS40
*BOMMP/SAVE/GO
EXIT
.
```

b) Version 2

```
. R LINK
*BOMMP (or MAIN2, SUB10, MACROS)
*(the 'REL' version of any user-defined subroutines)
*/SEARCH DUS10
*BOMMP/SAVE/GO
EXIT
.
```

c) Version 3

```
. R LINK
*MAIN3, SUB40, INT40, MACROS
*(the 'REL' versions of any user-defined subroutines)
*/SEARCH DUS40
*BOMMP/SAVE/GO
EXIT
.
```

d) Version 4

```
. R LINK
*MAIN4, SUB10, INT10, MACROS
*(the 'REL' versions of any user-defined subroutines)
*/SEARCH DUS10
*BOMMP/SAVE/GO
EXIT
.
```

2.4 Running BOMMP on Other Machines

In order to run BOMMP on other machines, the following routines, which are MACRO-10 coded, should be replaced by equivalent routines. In some cases, it is necessary to perform the same operation, while in others, it is sufficient to provide a dummy routine thereby removing the appropriate facility. The routines are described below (see Ref. 1 for details on facilities provided by routines) and, where a dummy replacement routine is possible, the replacement procedure is given.

a) SUBROUTINE EXPAND (NSIZE, ARRAY)

Description — Expands the size of array ARRAY to NSIZE elements.

Replacement — Use only Version 2 or 4.

b) LOGICAL FUNCTION FCHECK (NAME)

Description — Searches the user's disk area for the filename in NAME. FCHECK is .TRUE. if the file is found.

Replacement — For the first entry, set FCHECK to .FALSE. if running interactively, or to .TRUE. if running non-interactively (Batch mode). On subsequent entries, set FCHECK to .TRUE. .

c) SUBROUTINE CPU (TCPU)

Description — For the initial entry, TCPU is the CPU time in seconds since the beginning of the job. On subsequent entries, TCPU is the CPU time since the previous entry.

Replacement — Set TCPU equal to zero.

d) LOGICAL FUNCTION TTYCHK (DUMMY)

Description — DUMMY is a dummy parameter. TTYCHK is .TRUE. if an "↑" (ASCII code 136) has been typed at the terminal to halt model execution.

Replacement — Set TTYCHK to .FALSE. .

e) SUBROUTINE CHARSH (ANS, X, I)

Description — CHARSH operates on an ascii string of five characters contained in X, and rotates or shifts them depending on I. The result is returned in ANS. For I positive, rotation to the left occurs. For I negative, shifting to the right occurs with null characters entered from the left.

Replacement — Not possible without extensive recording.

f) SUBROUTINE RUN (PRONAM)

Description — Exits from BOMMP and then runs the program in PRONAM.

Replacement — Print on terminal that the 'PROGRAM' command is invalid (do not use the 'PROGRAM' command). Control automatically returns to the command mode.

When using the Adams integrator, some modifications are necessary to the FORTRAN coded subroutine STEP in the file INTEG.FOR. For the particular computer being used, the round-off error U should be determined: it is defined as the smallest value such that $(1{\cdot}0 + U)$ is greater than 1·0. The variables TWOU (i.e. $2 \times U$) and FOURU (i.e. $4 \times U$) should be set in the appropriate DATA statement. It should be noted that the subroutine STEP and the subroutine F, which is declared EXTERNAL in STEP, have different parameter lists to those in Reference 3.

In BOMMP, the DEC FORTRAN statements 'OPEN' and 'CLOSE', which perform basic file handling operations, may need to be replaced.

3. THE OUPUT PROGRAM 'TRANS'

3.1 Improvements

Most of the improvements described below have been implemented by using new commands. These are listed in Table 4 together with a reference to the appropriate section in this document.

TABLE 4
New TRANS Commands

Command	Reference
BARS	3.1.11
CYCLE	3.1.2
LIST	3.1.4
PRKONSTANT	3.1.5
REPEAT	3.1.9
SPACE	3.1.10
XBLOCK	3.1.7

3.1.1 *DEC FORTRAN-10 Compilation*

As with BOMMP, minor FORTRAN code changes have been made to allow compilation using the F10 compiler. Because of the resulting absence of the expandable array facility, when using this compiler, there is a limit of 300 blocks that can be processed for each input file. This limit can be altered by changing the appropriate DIMENSION statements in the main program of the file TRANSA.FOR (see Section 3.2).

3.1.2 Non-Interactive Running and Batch Processing

As with BOMMP, TRANS may now be run non-interactively, during On-Line or Batch processing. When running TRANS interactively, all commands and answers to prompts necessary for similar processing of other input files are written into the file TRANS.OUT. By renaming this file TRANS.IN, subsequent input files are processed non-interactively: instructions are read from the file TRANS.IN. The renaming may be achieved automatically while still running TRANS (i.e. before using the 'EXIT' command) by using a new command, 'CYCLE'. Non-interactive running requires only the filename to be specified, with a 'carriage-return' terminating program execution. To revert to interactive running, the file TRANS.IN must be renamed or deleted from the user's disk area. Because not all answers typed by the user are written into TRANS.OUT (e.g. when using the 'SCALE' or 'LABEL' commands), the user is advised against creating his own TRANS.IN file independently of running TRANS.

3.1.3 Preservation of Output Files

Assuming an input file named A.DAT, the output files created by TRANS for tabular or graphical output are A.COL, A.PPL, and A.PLT. If an input file with the same name (i.e. A.DAT) were subsequently processed, these output files would be overwritten or deleted when using the old version of TRANS. With the new version, the extension name is modified to allow preservation of up to 101 files for each type of output. For the above example, these files are:

A.COL, A.COn (for n = 0 . . . 9), A.Cnn (for nn = 10 . . . 99)

A.PPL, A.LPn (for n = 0 . . . 9), A.Lnn (for nn = 10 . . . 99)

A.PLT, A.PLn (for n = 0 . . . 9), A.Pnn (for nn = 10 . . . 99)

For more than 101 files, the output is appended to the last appropriate file (e.g. A.C99). Following the 'PRC', 'PRP', 'PLS', and 'PLO' commands, the output file is specified on the terminal if the particular output filename has been modified as above (for non-interactive runs as well). For example, after a 'PLS' or 'PLO' command when the files A.PLT . . . A.P11 already exist on the user's disk area, the following message is typed:

[PLS/O output, for this run, going to DSK:A .P12]

If all the files A.PLT . . . A.P99 exist, then the message typed is:

[PLS/O output, for this run, appended to DSK:A .P99]

3.1.4 Specifying Block Numbers

When specifying block numbers in response to the prompt:

BLKS

the user must now type "AV" (previously "A") to specify all blocks. In all instances, except when specifying output for the line printer using the 'PRC' or 'PRP' commands (see Sections 3.1.5 and 3.1.6), typing "A" now specifies only those blocks whose value varies.

The user may also specify a list, which consists of up to 30 block numbers, by typing "Ln", where n = 1 . . . 9. The 'LIST' command is used to define the composition of each list. All block number lists are stored in the file TRANS.BLK and are therefore available to further runs of TRANS.

3.1.5 Revised Operation of 'PRCOLUMN' Command

When specifying the blocks to be printed on the line printer, by typing "A", the constant values of the non-varying blocks are first printed followed by the varying block values in tabular form. By using the 'PRKONSTANT' command, only the constant values of the non-varying blocks are output for printing on the line printer.

3.1.6 Revised Operation of 'PRPLOT' Command

When specifying the blocks to be plotted on the line printer, by typing "A", only the varying blocks are plotted. Any non-varying blocks are printed before the plots as if the 'PRKONSTANT' command had first been used.

3.1.7 Cross Plots

The 'XBLOCK' command allows the X axis block to be redefined for incremental plots, thus enabling 'cross plots' of dependant variables.

3.1.8 Revised Form of 'Overlay' Plots

For 'overlay' plots, curves previously labelled by different symbols are now identified by different broken lines. This was found to give a less cluttered appearance to the plots.

3.1.9 Plotting Data from Different Input Files on the Same Graphs

The 'REPEAT' command is equivalent to the 'CYCLE' command except that, for 'strip' plots, subsequently processed files are plotted on the same graph(s). The different curves are identified by different broken lines. In the key for each graph, the broken lines are labelled with the titles (as they appear in the input files) and the input filenames. The title at the base of each graph is that stored by TRANS on executing the first 'GOE' command. The user is advised against using 'overlay' plots when using the 'REPEAT' command.

3.1.10 Spacing between Incremental Plots

Incremental plots are suitably spaced on 8·5 inch folding paper for the pen initially set at a fold. When plotting on non-folding paper, unnecessary spacing can be removed by using the 'SPACE' command.

3.1.11 Bars on Incremental Plots

The 'BARS' command results in the placement of bars on the incremental plot curves at specified equal independent variable intervals. The bars are short lines perpendicular to the curve and they may be used to provide a third demension, e.g. on a 'cross plot' of X and Y coordinates of an aircraft, the bar spacing will indicate speed if the independant variable is time.

3.2 Source Files

The source files for TRANS may be found on DECtape 068. Files with extensions 'FOR'

and 'MAC' are FORTRAN and MACRO coded respectively. The files are TRANSA.FOR, TRANSB.FOR, FCHECK.MAC, and SYMBOL.MAC.

3.3 Running TRANS on the ARL PDP-10 Computer

The source files are first compiled, using either the F10 or F40 compiler for the FORTRAN files. Then, assuming that the required 'REL' files are on the user's own disk area, a core image of the program may be produced in the file TRANS.SAV as follows. The files are loaded using the system program 'LINK'.

```
. R LINK

*TRANSA, TRANSB, FCHECK, SYMBOL

*TRANS/SAVE/GO

 EXIT

.
```

Generally, the above procedure is unnecessary, since a copy of TRANS.SAV may be found on DSKB:[1031, 1161].

3.4 Running TRANS on Other Machines

In order to run TRANS on other machines, the machine coded routines FCHECK, SYMBOL, and PLOT should be replaced by equivalent routines. A description and suggested replacement of FCHECK is given in Section 2.4. The routines SYMBOL and PLOT are used extensively in generating incremental plotter output: replacement routines are therefore essential if incremental plots are required. The descriptions of these two routines are given below.

a) SUBROUTINE SYMBOL (LU, X, Y, HEIGHT, ASCII, THETA, N)

The subroutine SYMBOL plots the ascii string of N characters contained in ASCII at the point X, Y. LU is the logical unit on which the plot file is being written. HEIGHT is the height (in inches) of the characters and THETA is the angle (in degrees) to the x-axis at which the string is to be plotted.

b) SUBROUTINE PLOT (LU, X, Y, N)

The subroutine PLOT controls the pen on the incremental plotter. LU is the same as in SYMBOL. X and Y are coordinates if they are real variables, or the number of 0·01 inch steps moved in the X and Y directions if they are integer. X and Y must be real for N = 1 or 2. For N = 1, X and Y are scaling factors. For N = 2, a new origin is defined such that X and Y are the coordinates of the current position of the pen with respect to the new origin. For N = 3, the pen is lifted, moved to X, Y and dropped. For N = 4, the pen is dropped and moved to X, Y. For N = 5, the pen is lifted, moved to X, Y but not dropped.

As with BOMMP, the DEC FORTRAN statements 'OPEN' and 'CLOSE' may need to be replaced.

4. CONCLUDING REMARKS

This postscript to previous publications on the simulation language CSMP-10(ARL) consists of a description of improvements, details of the location of source files and their use on the ARL computer, and suggestions on how to implement the language on other computers. The improvements significantly increase the efficiency of the language in handling simulation problems. DEC FORTRAN-10 compilation and an alternative integration method that is very efficient for certain systems make possible large savings in CPU time. The many improvements to the output program enable it to be used with much greater flexibility, especially when handling several output files concurrently, e.g. plotting data from different files on the same graph. The aircraft arresting gear problem of Reference 4 is used in a comprehensive demonstration of the new operating procedures and improvements to the simulation language.

REFERENCES

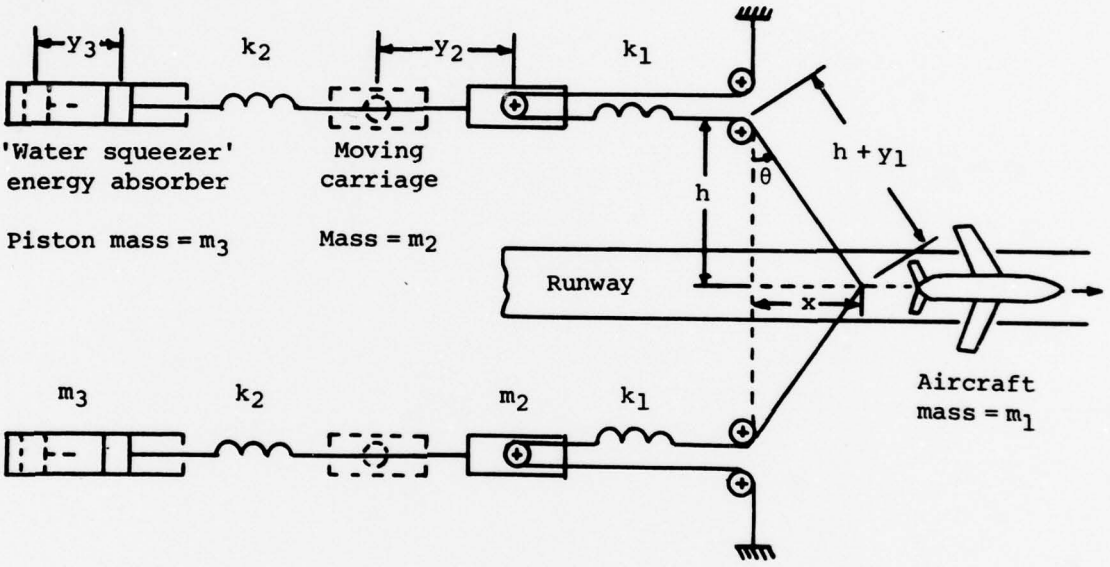
1. Gilbert, N. E. and Nankivell, P. G. — 'The Simulation Language CSMP-10(ARL)'. ARL Note Aero 362, May 1976.
2. Nankivell, P. G. and Gilbert, N. E. — 'A General Purpose Output Program for Use in Simulation'. ARL Note Aero 367, December 1976.
3. Gilbert, N. E. and Nankivell, P. G. — 'An Extended Version of CSMP-10'. Simulation, Vol. 29, No. 2, August 1977.
4. Harnett, R. J. and Sansom, F. J. — 'MIDAS Programming Guide'. Report No. SEG-TDR-64-1, Wright Patterson Air Force Base, Ohio, U.S.A., 1964.
5. Shampine, L. F. and Gordon, M. K. — 'Computer Solution of Ordinary Differential Equations'. W. H. Freeman and Company, San Francisco, California, 1975.

APPENDIX A

Use of CSMP-10(ARL) for the Aircraft Arresting Gear Problem

A.1 Description of Problem

(1) *Physical diagram*



(2) *Equations*

$$m_3 \ddot{y}_3 = f_{k2} - f_D \qquad y_3(0) = \dot{y}_3(0) = 0$$

$$m_2 \ddot{y}_2 = 2f_{k1} - f_{k2} \qquad y_2(0) = \dot{y}_2(0) = 0$$

$$m_1 \ddot{x} = -2f_{k1} \sin\theta \qquad x(0) = 0,\ \dot{x}(0) \text{ is variable}$$

$$f_{k2} = k_2 (y_2 - y_3) \quad \text{if } y_3 < y_2$$

$$= 0 \quad \text{if } y_3 \geq y_2$$

$$f_{k1} = k_1 (y_1 - 2y_2) \quad \text{if } y_1 > 2y_2$$

$$= 0 \quad \text{if } y_1 \leq 2y_2$$

$$f_D = f(y_3)\, y_3^2 \quad \text{where } f(y_3) \text{ is defined in (4)}$$

$$y_1 = (x^2 + h^2)^{\frac{1}{2}} - h$$

$$\sin\theta = x/(h + y_1)$$

(3) *Constants*

$m_1 = 1400$ slug
$m_2 = 45 \cdot 28$ slug
$m_3 = 20$ slug
$k_1 = 4550$ lb/ft
$k_2 = 25300$ lb/ft
$h = 125$ ft

(4) *Function* $f(y_3)$

Defined by linear interpolation of the following coordinates:

y_3 (ft)	$f(y_3)$ (lb/(ft/s)²)
0	8·33
30	4·0
60	1·6
120	5·2
150	5·2
180	6·6
210	8·3
240	10·7
270	16·0
282	21·0
294	28·0
306	41·0
312	50·0
324	90·0
999	90·0

(5) *Output*

a) At $\dot{x}(0) = 290$ ft/s using both Adams and Runge-Kutta integrators for time $t = 0$ to 10s,

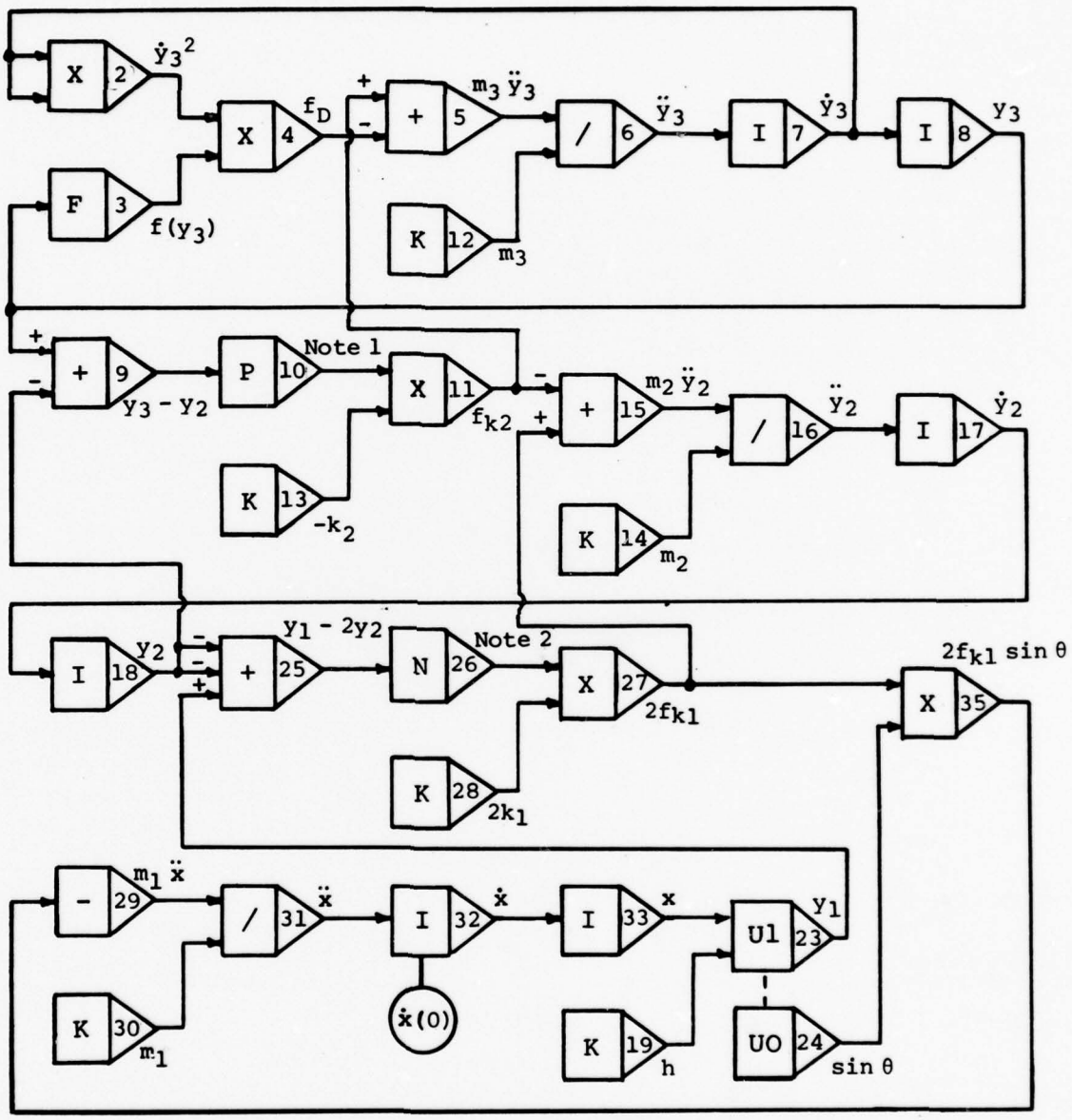
(i) tabular output of $\dot{y}_3$, y_3, f_D, y_2, $\dot{x}$, x in 1s intervals;
(ii) graphical output* of $\dot{x}$, x, y_3 vs. t; and
(iii) graphical output* of x vs. $\dot{x}$.

b) Graphical output of $\dot{x}$, x, y_3 vs. t for $t = 0$ to 10s for $\dot{x}_0 = 290$, 250, 210 ft/s (on same graph) using Adams integrator.

---

* Since graphs are almost identical for runs with either integrator used (assuming the same solution accuracy), both graphs are not shown below.

(6) *Block diagram* (unless shown, integrator initial conditions are zero; see Fig. 1 of Ref. 1 for notation)



Notes:

1. Block 10 output $= y_3 - y_2$ if $y_3 < y_2$
   $= 0$ if $y_3 \geq y_2$
2. Block 26 output $= y_1 - 2y_2$ if $y_1 > 2y_2$
   $= 0$ if $y_1 \leq 2y_2$

A.2 Model Input File

```
CONFIGURATIONS
    X DOT(0) = 290
    2   X     7   7
    3   F     8
    4   X     2   3      ;    FD
    5   +    11  -4
    6   /     5  12      ;Y3 DOT DOT
    7   I     6          ;  Y3 DOT
    8   I     7          ;    Y3
    9   +     8 -18
   10   P     9
   11   X    10  13      ;    FK2
   12   K                ;    M3
   13   K                ;   -K2
   14   K                ;    M2
   15   +   -11  27
   16   /    15  14      ;Y2 DOT DOT
   17   I    16          ;  Y2 DOT
   18   I    17          ;    Y2
   19   K                ;    H
   23   U1   33  19      ;    Y1
   24   U0   23          ;SIN(THETA)
   25   +   -18 -18  23
   26   N    25
   27   X    26  28      ;   2FK1
   28   K                ;   2K1
   29   -    35
   30   K                ;    M1
   31   /    29  30      ;X DOT DOT
   32   I    31          ;  X DOT
   33   I    32          ;    X
   35   X    27  24      ;2FK1*SIN
```

```
PARAMETERS
   12  2.0000E+01
   13 -2.5300E+04
   14  4.5280E+01
   19  1.2500E+02
   28  9.1000E+03
   30  1.4000E+03
   32  2.9000E+02

FUNCTIONS
    3
  0.0000E-01  8.3300E+00
  3.0000E+01  4.0000E+01
  6.0000E+01  1.6000E+01
  1.2000E+02  5.2000E+01
  1.5000E+02  5.2000E+00
  1.8000E+02  6.6000E+00
  2.1000E+02  8.3000E+00
  2.4000E+02  1.0700E+01
  2.7000E+02  1.6000E+01
  2.8200E+02  2.1000E+01
  2.9400E+02  2.8000E+01
  3.0600E+02  4.1000E+01
  3.1200E+02  5.0000E+01
  3.2400E+02  9.0000E+01
  9.9900E+02  9.0000E+01
```

A.3 User-defined Subroutine

```
      SUBROUTINE USER1(IB,C,MTRX,PAR)

C     USER-DEFINED SUBROUTINE FOR
C     THE ARRESTING GEAR PROBLEM
C
C     IT CALCULATES
C           Y1=SQRT(X*X+H*H)-H
C     AS ITS PRINCIPLE OUTPUT, AND
C           SIN(THETA)=X/(H+Y1)
C     AS AN ADDITIONAL OUTPUT VIA A UO BLOCK.

      DIMENSION C(1),MTRX(1),PAR(1)
      COMMON /TEST/TEST(9)
      INTEGER TEST
      DATA IBUO1/24/

C     FOR INITIAL EXECUTION TEST(5)=1

      IF(TEST(5).NE.1) GO TO 10

C     GET INPUT BLOCK NUMBERS

      IB1=MTRX(5*(IB-1)+2)
      IB2=MTRX(5*(IB-1)+3)

C     GET INPUT VALUES

10    X=C(IB1)
      H=C(IB2)

C     CALCULATE

      Y1=SQRT(X*X+H*H)-H
      STHETA=X/(H+Y1)

C     SET OUTPUT VALUES

      C(IB)=Y1
      C(IBUO1)=STHETA

      RETURN
      END
```

A.4 Creating and Running Modelling Program

(1) *Creating stored program using Version 4 of BOMMP*

```
.COMPILE/F10/NEW EXUSER.FOR
FORTRAN: EXUSER
USER1

.R LINK

*MAIN4,SUB10,INT10,MACROS
*EXUSER
*/SEARCH DUS10
*ARSTR/SAVE/GO

EXIT

.
```

(2) *Running program with Adams integrator for* $\dot{x}_0 = 290, 250, 210$
(uses default Adams integration parameters)

```
.RU ARSTR


MAX BLK NO. = 300

MAX NO. OF: I & T1 BLKS, U BLKS, F BLKS = 100,25,25

INTEGRATORS:
   RUNGE-KUTTA 2ND ORDER [DEFAULT]
   ADAMS VARIABLE STEP (M.K.GORDON IMPLEMENTATION)


*LOG8:ARSTR<I

MODEL I/P FROM  LOG8:ARSTR.MOD

*LOG9:OUT1<O

BLOCK O/P   TO  LOG9:OUT1 .DAT
```

```
*CON

CONFIGURATIONS :

    X DOT(0) = 290

 BLK  TYPE  B1  B2  B3
   2   X     7   7
   3   F     8
   4   X    +0


*PAR

PARAMETERS :

 BLK      P1          P2          P3
  12  2.0000E+01
  13 -2.5300E+04+0


*FUN

FUNCTIONS :

BLK NO.   3

COORD PAIRS :

 0.0000E+00  8.3300E+00
 3.0000E+01  4.0000E+00+0


*INT

INTEGN PARAMS; LOWER, UPPER, INTERVAL = 0,10,0


ADAMS INTEGRATOR : Y

*OUT

O/P BLKS
A

O/P PARAMS; % CHANGE REQRD, INTERVAL = 0.0001,0.1

*GOE

** RUNNING **
```

```
RUN CPU TIME : 15.78 SEC.

*TIT

TITLE (LIMIT 60 CHRS)
    X DOT(0) = 250

*PAR

PARAMETERS :

BLK, P1, P2, P3
32,250

*LOG9:OUT2<0

BLOCK O/P   TO  LOG9:OUT2 .DAT

*GOE

** RUNNING **

RUN CPU TIME : 15.07 SEC.

*TIT

TITLE (LIMIT 60 CHRS)
    X DOT(0) = 210

*PAR

PARAMETERS :

BLK, P1, P2, P3
32,210

*LOG9:OUT3<0

BLOCK O/P   TO  LOG9:OUT3 .DAT

*GOE

** RUNNING **

RUN CPU TIME : 13.64 SEC.

*EXI


END OF EXECUTION
CPU TIME: 49.18 ELAPSED TIME: 7:29.60
EXIT

.
```

(3) *Preparing for further runs that are partly non-interactive*

```
.R PIP

*BOMMP.IN=BOMMP.OUT/T
*↑C

.TYPE BOMMP.IN
LOG8:ARSTR←I
LOG9:OUT1←O
CON

PAR

FUN

INT
0,10,0
Y
OUT
A
0.0001,0.1
GOE
TIT
    X DOT(0) = 250
PAR
32,250

LOG9:OUT2←O
GOE
TIT
    X DOT(0) = 210
PAR
32,210

LOG9:OUT3←O
GOE
EXI

.TECO BOMMP.IN

[2K CORE]
*SINT$$LT$$
INT
*30KT$$
*IMAN
$-3T$$
FUN

MAN
*EX$$
```

```
.TYPE BONMP.IN
LOG8:ARSTR←I
LOG9:OUT1←O
CON

PAR

FUN

MAN

.
```

(4) *Running program with Runge Kutta integrator for* $\dot{x}_0 = 290$
(program is run partly non-interactive; shows use of (i) 'RESUME' following "↑" and (ii) 'RETAIN'

```
.RU ARSTR


MAX BLK NO. = 300

MAX NO. OF: I & T1 BLKS, U BLKS, F BLKS = 100,25,25

INTEGRATORS:
   RUNGE-KUTTA 2ND ORDER [DEFAULT]
   ADAMS VARIABLE STEP (M.K.GORDON IMPLEMENTATION)


    X DOT(0) = 290

*LOG9:OUTRK←O

BLOCK O/P   TO  LOG9:OUTRK.DAT

*INT

INTEGN PARAMS; LOWER, UPPER, INTERVAL = 0,10,0.005


ADAMS INTEGRATOR : N

*OUT

O/P BLKS
A

O/P PARAMS; Z CHANGE REQRD, INTERVAL = 0.0001,0.1

*GOE
```

```
** RUNNING **^

RUN CPU TIME :  6.68 SEC.

RUN TERMINATED BY A "↑"


*LOO

BLK = 1
O/P =  2.6000E+00

BLK = 32
O/P =  1.8303E+02

BLK =

*RES

** RUNNING **

RUN CPU TIME : 25.94 SEC.

*LOO

BLK = 1
O/P =  1.0000E+01

BLK = 32
O/P =  5.9902E+00

BLK =

*RET

*LOG10:FINSH←M

MODEL O/P   TO LOG10:FINSH.MOD

*STO

CON, PAR, FUN, OR ALL :ALL

*TTY:←M

MODEL O/P   TO   TTY:

*STO
```

```
CON, PAR, FUN, OR ALL :PAR

   7  2.9794E+00
   8  3.8383E+02
  12  2.0000E+01
  13 -2.5300E+04
  14  4.5289E+01
  17  2.9736E+00
  18  3.8386E+02
  19  1.2500E+02
  24  9.9015E-01
  28  9.1000E+03
  30  1.4000E+03
  32  5.9902E+00
  33  8.8401E+02

*EXI


END OF EXECUTION
CPU TIME: 30.14 ELAPSED TIME: 5:17.78
EXIT

.
```

A.5 Running Output Program

(1) *Output of corresponding runs for* $\dot{x}_0 = 290$ *using Adams and Runge-Kutta integrators* (shows use of 'LIST', 'PRKONSTANT', 'SPACE', 'XBLOCK', 'BARS', and 'CYCLE'; files TRANS.IN and TRANS.BLK, as well as tabular and incremental plotter output, are presented)

```
.R PIP

*/X+[1031,1161]TRANS.SAV
*↑C

.RU TRANS


[TRANS VERSION DATE 24-FEB-78]

I/P FILENAME = OUT1

    X DOT(0) = 290

I/P FILE RECORDED ON 27-FEB-78 AT 11:16

INTEGN INT = 0.0000E+00; RUN CPU TIME = 15.78 SEC.

TIME FROM 0.0000E+00 TO 1.0000E+01 IN STEPS OF 1.0000E-01

*LIS

BLOCK LIST NUMBER = 1

LIST OF BLOCK NUMBERS = 7,8,4,18,32,33

BLOCK LIST NUMBER = 2

LIST OF BLOCK NUMBERS = 32,33,7

BLOCK LIST NUMBER =

*TIM

TIME PARAMS; LOWER, UPPER, INTERVAL = 0,10,1

*PRK

*PRC

PRINTING IN COLUMNS :

BLKS
L1

IS O/P TO TTY REQRD : N

*SPA

IS SPACING BETWEEN PLOTS REQRD : N
```

```
*PLS

STRIP PLOTS :

BLKS
L2

LENGTH OF AXES IN INCHES; X, Y = 5,2

*GOE

** RUNNING **


*XBL

NEW X AXIS BLK NO. = 32

*BAR

INDEP VARIABLE INTERVAL BETWEEN BARS = 1

*PLS

STRIP PLOTS :

BLKS
33

LENGTH OF AXES IN INCHES; X, Y = 5,5

*GOE

** RUNNING **


*CYC

I/P FILENAME = OUTRK


** RUNNING **


** RUNNING **


I/P FILENAME =


END OF EXECUTION
CPU TIME: 16.34 ELAPSED TIME: 3:34.62
EXIT
```

```
.TYPE TRANS.IN
LIS
TIM
 0.0000E+00 1.0000E+01 1.0000E+00
PRK
PRC
L1

N
SPA
N
PLS
L2

  5.0  2.0
GOE
XBL
  32
BAR
 1.0000E+00
PLS
33

  5.0  5.0
GOE
CYC

.DELETE TRANS.IN
FILES DELETED:
TRANS.IN
01 BLOCKS FREED

.TYPE TRANS.BLK
L1
   7   8   4  18  32  33   0   0   0   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0   0   0   0
L2
  32  33   7   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0   0   0   0

.R PRINT


11:36*OUT1.COL/P,OUTRK.COL/P$
FILES PRINTED   REQ.    PAGES

DSK:OUT1.COL    R01     2
DSK:OUTRK.COL   R02     2
EXIT

.R PLOTQ


11:37*OUTRK.PLT$
FILES PLOTTED   REQ.    FEET    LIMITS
OUTRK   .PLT    R01     2       X(3.28",1'7"),Y(-5.32",5")
EXIT

.
```

Tabular output file OUT1.COL

(uses Adams integrator)

```
    X DOT(0) = 290

RECORDED ON 27-Feb-78 AT 11:16   INTEGN INT = 0.0000E+00  RUN CPU TIME = 15.78 SEC.

NON-VARYING BLOCK VALUES

BLK NUMBER =     12          13          14          19          28          30
                 M3          -K2         M2          H           2K1         M1

             2.0000E+01 -2.5300E+04 4.5280E+01 1.2500E+02 9.1000E+03 1.4000E+03
```

```
    X DOT(0) = 290

RECORDED ON 27-Feb-78 AT 11:16   INTEGN INT = 0.0000E+00  RUN CPU TIME = 15.78 SEC.

BLK NUMBER =      7           8           4           18          32          33
   TIME        Y3 DOT        Y3          FD          Y2        X DOT         X

0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  2.9000E+02  0.0000E+00
1.0000E+00  1.0002E+02  8.7530E+01  3.7942E+04  8.9114E+01  2.7022E+02  2.8151E+02
2.0000E+00  1.0512E+02  2.0153E+02  8.6415E+04  2.0493E+02  2.2205E+02  5.2966E+02
3.0000E+00  6.9323E+01  2.9070E+02  1.2532E+05  2.9559E+02  1.5190E+02  7.1871E+02
4.0000E+00  3.8730E+01  3.4032E+02  1.3500E+05  3.4559E+02  5.0044E+01  8.2115E+02
5.0000E+00  7.6788E+00  3.6355E+02  5.3068E+03  3.6375E+02  1.1254E+01  8.4378E+02
6.0000E+00  4.7752E+00  3.6890E+02  2.0522E+03  3.6898E+02  9.5154E+00  8.5409E+02
7.0000E+00  4.1589E+00  3.7334E+02  1.5507E+03  3.7340E+02  8.2895E+00  8.6296E+02
8.0000E+00  3.6611E+00  3.7723E+02  1.2063E+03  3.7728E+02  7.3470E+00  8.7076E+02
9.0000E+00  3.2823E+00  3.8070E+02  9.6963E+02  3.8074E+02  6.5988E+00  8.7772E+02
1.0000E+01  2.9648E+00  3.8383E+02  7.9111E+02  3.8386E+02  5.9899E+00  8.8400E+02
```

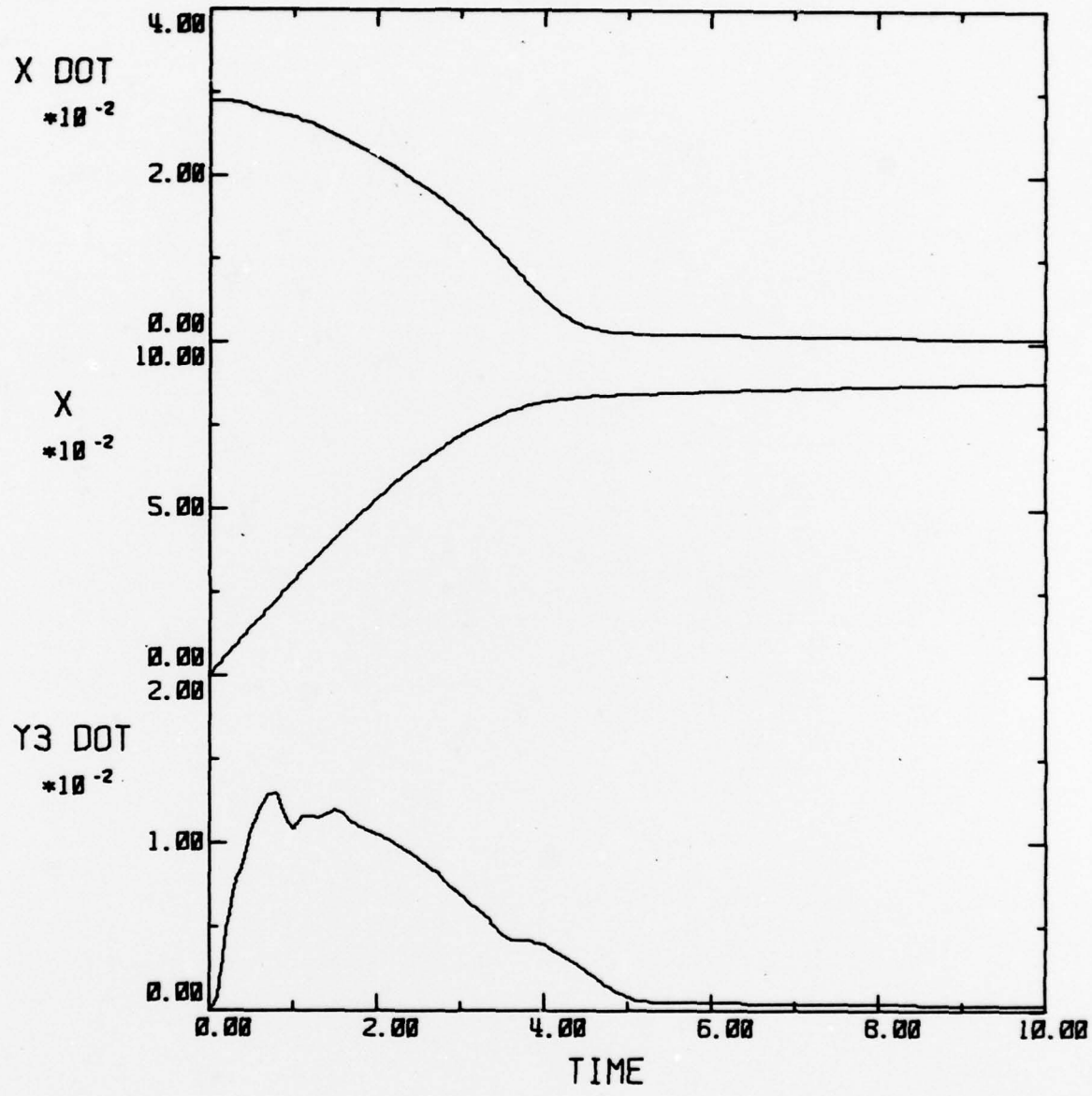
Tabular output file OUTRK.COL

(uses Runge-Kutta integrator)

```
    X DOT(0) = 290

RECORDED ON 27-Feb-78 AT 11:25   INTEGN INT = 5.0000E-03  RUN CPU TIME = 25.94 SEC.

NON-VARYING BLOCK VALUES

BLK NUMBER =     12          13          14          19          28          30
                 M3          -K2         M2          H           2K1         M1

             2.0000E+01 -2.5300E+04 4.5280E+01 1.2500E+02 9.1000E+03 1.4000E+03



    X DOT(0) = 290

RECORDED ON 27-Feb-78 AT 11:25   INTEGN INT = 5.0000E-03  RUN CPU TIME = 25.94 SEC.

BLK NUMBER =     7           8           4           18          32          33
   TIME       Y3 DOT        Y3          FD          Y2        X DOT         X

0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  2.9000E+02  0.0000E+00
1.0000E+00  1.0020E+02  8.7526E+01  3.8964E+04  8.9113E+01  2.7022E+02  2.8151E+02
2.0000E+00  1.0598E+02  2.0154E+02  8.6352E+04  2.0493E+02  2.2205E+02  5.2966E+02
3.0000E+00  6.9033E+01  2.9070E+02  1.2427E+05  2.9559E+02  1.5190E+02  7.1870E+02
4.0000E+00  3.8562E+01  3.4032E+02  1.3383E+05  3.4559E+02  5.0047E+01  8.2115E+02
5.0000E+00  7.7659E+00  3.6355E+02  5.4266E+03  3.6375E+02  1.1255E+01  8.4378E+02
6.0000E+00  4.7656E+00  3.6890E+02  2.0440E+03  3.6898E+02  9.5161E+00  8.5489E+02
7.0000E+00  4.1396E+00  3.7334E+02  1.5423E+03  3.7340E+02  8.2900E+00  8.6296E+02
8.0000E+00  3.6623E+00  3.7723E+02  1.2071E+03  3.7728E+02  7.3473E+00  8.7076E+02
9.0000E+00  3.2852E+00  3.8079E+02  9.7133E+02  3.8074E+02  6.5991E+00  8.7772E+02
1.0000E+01  2.9794E+00  3.8383E+02  7.9893E+02  3.8386E+02  5.9902E+00  8.8401E+02
```

Graphical output file OUTRK.PLT

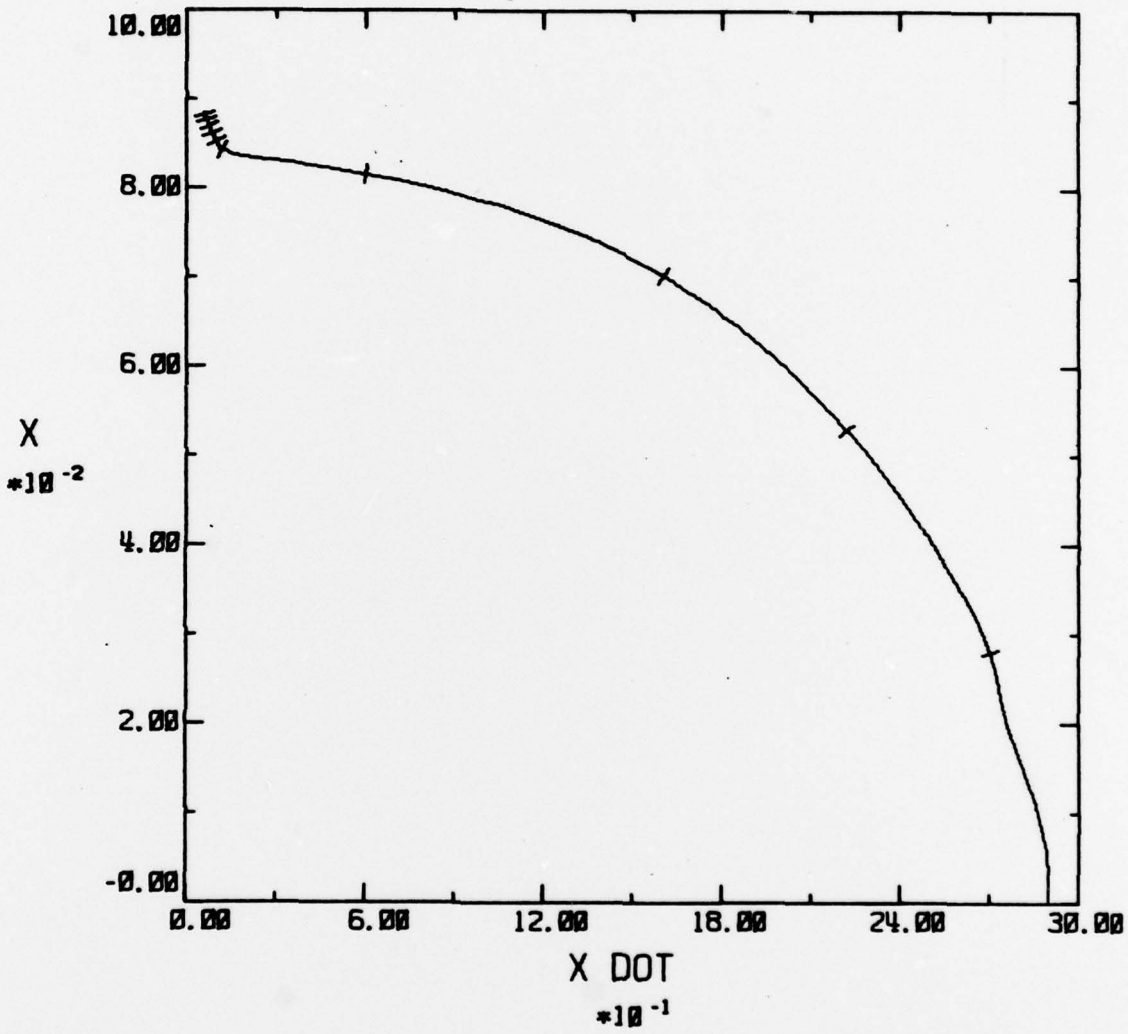
(a) strip plots with time as abscissa



X DOT(0) = 290

Graphical output file OUTRK.PLT

(b) cross plot of x vs. $\dot{x}$



X DOT(0) = 290

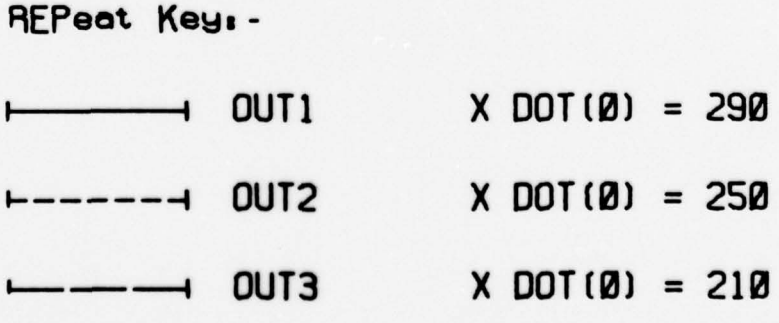
(2) *Plotting three runs performed in Section* (2) *of Appendix A.4 on same graph*
(shows use of 'REPEAT' and preservation of output files)

```
.RU TRANS

[TRANS VERSION DATE 24-FEB-78]

I/P FILENAME = OUT1

    X DOT(0) = 290

I/P FILE RECORDED ON 27-FEB-78 AT 11:16

INTEGN INT = 0.0000E+00; RUN CPU TIME = 15.78 SEC.

TIME FROM 0.0000E+00 TO 1.0000E+01 IN STEPS OF 1.0000E-01

*PLS

[PLS/O OUTPUT, FOR THIS RUN, GOING TO DSK:OUT1 .PL0]

STRIP PLOTS :

BLKS
L2

LENGTH OF AXES IN INCHES; X, Y = 5,2

*LAB

BLK NO. -1 DENOTES INDEP VARIABLE

IS LABELLING TO BE READ FROM DSK : N

IS TTY LISTING REQRD : N

ARE MODIFICATIONS REQRD : Y

BLK (0 FOR TITLE), LINE, TEXT
0,1,          AIRCRAFT ARRESTING GEAR PROBLEM

*GOE

** RUNNING **


*REP

I/P FILENAME = OUT2

** RUNNING **
```

```
I/P FILENAME = OUT3

** RUNNING **


I/P FILENAME =


END OF EXECUTION
CPU TIME: 8.96  ELAPSED TIME: 3:9.40
EXIT

.TYPE TRANS.IN
PLS
L2

  5.0  2.0
LAB
GOE
REP

.R PLOTQ


11:41*OUT1.PL0$
FILES PLOTTED   REQ.    FEET    LIMITS
OUT1    .PL0    R01     2       X(8.50",1'9.24"),Y(-5.32",5")
EXIT

.
```

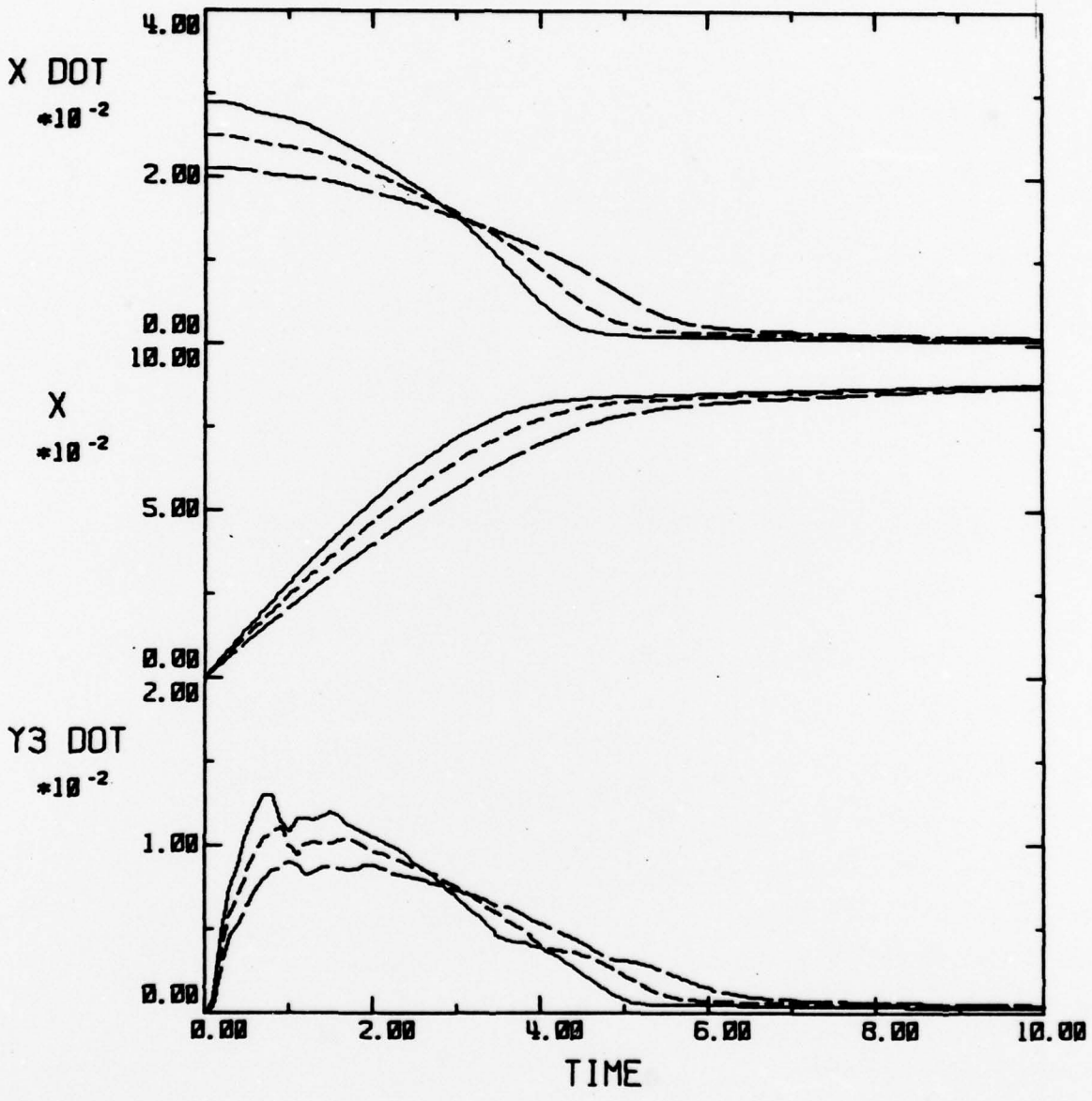
Graphical output file OUT1.PL0

(a) 'REPEAT' key (appears to right of strip plots on incremental plotter output)



Graphical output file OUT1.PL0

(b) strip plots



AIRCRAFT ARRESTING GEAR PROBLEM

DOCUMENT CONTROL DATA SHEET

Security classification of this page: Unclassified

1. Document Numbers
   (a) AR Number: AR-001-267
   (b) Document Series and Number: Aerodynamics Note 375
   (c) Report Number: ARL-Aero-Tech.-Note-375

2. Security Classification
   (a) Complete document: Unclassified
   (b) Title in isolation: Unclassified
   (c) Summary in isolation: Unclassified

3. Title: FURTHER INFORMATION FOR USERS OF THE SIMULATION LANGUAGE CSMP-10(ARL)

4. Personal Author(s):
   N. E. Gilbert and
   P. G. Nankivell

5. Document Date:
   May, 1978

6. Type of Report and Period Covered:
   Tech. Note

7. Corporate Author(s):
   Aeronautical Research Laboratories

8. Reference Numbers
   (a) Task: NAV 74/4
   (b) Sponsoring Agency: ARL File A2/19

9. Cost Code:
   51 7740

10. *Imprint:*
    Aeronautical Research Laboratories,
    Melbourne, 1978

11. *Computer Program(s)*
    (Title(s) and language(s)):
    BOMMP, TRANS (FORTRAN IV)

12. Release Limitations (of the document):
    Approved for public release

12-0. Overseas:	No.		P.R.	1	A		B		C		D		E	

13. Announcement Limitations (of the information of this page):
    No limitation

14. Descriptors:
    Simulation languages, Simulation, Mathematical models, Computerized simulation, CSMP (Computer language)

15. Cosati Codes:
    0902, 1201, 1402

16.

ABSTRACT

This document is a postscript to previous publications on the simulation language CSMP-10(ARL). It consists of a description of improvements, details of the location of source files and their use on the ARL computer, and suggestions on how to implement the language on other computers.

DISTRIBUTION

AUSTRALIA

DEPARTMENT OF DEFENCE	Copy No.
Central Office	
Chief Defence Scientist	1
Executive Controller, ADSS	2
Superintendent, Defence Science Administration	3
Aust. Defence Scientific and Technical Representative (U.K.)	4
Counsellor, Defence Science (U.S.A.)	5
Defence Library	6
JIO	7
Assistant Secretary, DISB	8–23
Aeronautical Research Laboratories	
Chief Superintendent	24
Superintendent, Aerodynamics Division	25
Divisional File, Aerodynamics Division	26
Authors: N. E. Gilbert	27–28
P. G. Nankivell	29–30
Library	31
L. H. Mitchell	32
D. A. Secomb	33–38
D. C. Collis	39–46
Materials Research Laboratories	
Library	47
Defence Research Centre	
Library	48
T. J. Packer, Aerospace Division	49
Central Studies Establishment Information Centre	
Library	50
Engineering Development Establishment	
Library	51
RAN Research Laboratory	
Library	52
Navy Office	
Naval Scientific Adviser	53
Army Office	
Army Scientific Adviser	54
Royal Military College	55
U.S. Army Standardisation Group	56
Air Office	
Air Force Scientific Adviser	57
Aircraft Research and Development Unit	58
Engineering (CAFTS) Library	59

UNITED STATES OF AMERICA

UNIVERSITIES AND COLLEGES